WORKING PAPER
ALFRED P. SLOAN SCHOOL OF MANAGEMENT

Associative Learning Project--Phase 1 System

by

John F. Rockart
Michael S. Scott Morton
Zenon S. Zannetos

January 1970 440-70

MASSACHUSETTS
INSTITUTE OF TECHNOLOGY
50 MEMORIAL DRIVE
CAMBRIDGE, MASSACHUSETTS 02139

Associative Learning Project--Phase 1 System

by

John F. Rockart
Michael S. Scott Morton
Zenon S. Zannetos

January 1970

440-70

Associative Learning Project--Phase 1 System

by

John F. Rockart
Michael S. Scott Morton
Zenon S. Zannetos

During the past eighteen months, the Management Planning Information and Control group at the Sloan School has been developing and experimenting with a new approach to computer assisted instruction. Built around a computer-driven interactive terminal linked to an "associative" memory with flexible search procedures, the system is aimed at providing (a) solutions to some critical problems that we believe are evident today in the field of education, and (b) a research tool for gaining insights into the nature of the learning process itself.

The purpose of this paper is to consider both the objectives and the techniques of the project. The emphasis, however, will be on the latter as a complete description of the philosophy which underlies the current work is available in a recent publication (1).

The following paragraphs will discuss, in order, the basic objectives of the present system, the characteristics necessary for the next generation of learning systems, the operational objectives of the present system, the hardware which is being utilized, and the basic structure of the software which comprises the associative learning system itself. The emphasis throughout this paper has been placed on allowing the reader to understand what is being done and why it is being done rather than on explaining the programming details of the system.

I. Basic Objectives

The system has four basic objectives. These are:

1. To allow the student to utilize his own particular background and to learn in his own way at his own particular rate of speed. Some, perhaps even a majority of, students desire to follow a carefully programmed path to learn a subject. They prefer that the professor determine not only the objectives of the material but also the method of learning itself. For these students the traditional programmed instruction techniques are effective, given some pre-prescribed branching procedure. In contrast, however, other students find most professor-directed learning material either inadequate or stultifying or both.

   In such cases the methods of association used by the students and professors are normally different. To ascertain that learning takes place in all cases, the teacher uses redundancy, and a variety of similes and analogies. This practice, however, does not aid the learning of the brighter students who do not need the repetition.

   For this last group of students it is necessary to develop a system which will allow the student to direct his own learning processes and find his own path to understanding. In other words the system

must allow the student to be in control of the learning process. As such programmed instruction must allow the student to switch from a highly structured material to an interrogative mode when he finds questions of particular interest to him. In addition, it is necessary eventually to provide the student with a range of models and subroutine "tools" to allow him to solve specific problems himself in an interactive learning mode. This system capability will allow the student to experiment with different methods of problem solution and will also provide him with immediate feedback in the form of results.

2. <u>To reduce the burden upon the teacher of presenting simple material repetitively term after term</u>. In every field today there is an ever-increasing collection of well-structured basic material on fundamentals which must be "taught" and learned by students within the field. To most professors, the timeworn repetitive rehashing of this material for each new class is boring in the extreme. Ideally we would like to teach in class only the material which is on the boundary of our understanding for it is this material which is most interesting to us and is apt to be most inspiring to a class of students.

   To enable professors and students to concentrate on the more intriguing material which is closer to the state-of-the-art, a means must be found to present effectively

the more basic and prerequisite material. Fortunately the basic fundamentals, the well-organized aspects of our understanding, are just those facets of the material which are easiest to assign to an inanimate teaching system. Furthermore, there is evidence that for repetitive tasks where learning has been exhausted, inanimate systems are more consistent and rational (because of their infinite patience), in most cases faster, do not require as much redundancy for purpose of communication (because of their capacity to reproduce accurately the desirable behavior) and allow decentralized control.

3. To allow the integration of instructional material within and across functional fields. If we examine the management of our educational efforts, we find that in spite of the fact that we offer training leading to our professional degrees, our curricula are structured in such a fashion that we teach in rigidly defined compartments. Very little effort is devoted to integrating the material in a student's major area of concentration, and even less to integrating across fields. The absence of such integration, or a framework which will enable the student to achieve integration, is in our opinion very detrimental for learning. It not only wastes valuable student and faculty time but also inhibits discovery. Assistance is needed in this area. It is an objective of the project to develop a tool which can

assist in integrating parts of knowledge, rather than even more rigidly isolating small chunks of knowledge. We are convinced that a success in this area will have a very profound impact on the process of learning because the latter is basically an associative and integrating process.

4. Provide an experimental setting for research in learning processes. At present we do not understand very well how learning occurs nor do we have dependable models to use for inference. Our project is aimed at providing us with empirical evidence on how students derive lower level educational objectives, and how they plan to reach them, (i.e., the means to ends chains which they choose). A study of the students' search paths will provide us, we believe, with evidence amenable to generalization which in turn will eventually lead to an understanding of the differences between successful and unsuccessful planning and search paths.

   Ancillary by-products of this effort will be:

   (a) Instructional systems with "guided-search" capabilities, to facilitate student learning.

   (b) Methods for indexing educational material so that it can be associated and retrieved easily, given a specification of the desired context.

II. Necessary Characteristics of the Next Generation of Learning Systems

The development of a system which will help all students to learn in their own way requires, we believe, four main underlying systems characteristics. These are:

A. Semantic content association

The first of these is the use of a semantic memory, with associations based on the content of information, not its location. This characteristic provides the necessary flexibility which allows the student to address the system in natural language and to follow paths through the material that are relevant to his line of questioning. Semantic-content association facilitates the introduction of new material and the updating and integration of such into the old material.

The design and use of a semantic memory implies a certain formal structuring of the information content, at least initially. Unfortunately there has not been enough development in our understanding of the structure of natural language to allow complete freedom to the user. However with structured material this is not a serious limitation as there is a "natural" formality to the subject matter which does not impose any severe limitations on the user. The vocabulary which the computer must "know" for many subject areas is, we believe, quite limited. The recognition of a selected set of key words has proven to be a sufficient base from which the system can initially be developed.

B. Capability for Learning

The second major distinguishing characteristic of our system is the importance we have attached to the system's "ability to learn." It is central to our notions that we build a system that is able to monitor the user. Thus, initially, we have sought to provide output which will enable us to recognize patterns in student learning behavior, and to adapt the system accordingly to better serve students. Ultimately, we feel that the system can be made to "learn" itself.

C. A "Responsive" Computer System

The "tutor" with whom the student is in contact must, we believe, possess as many of the characteristics of the human tutor as possible. There must be no inflexibility due to awkwardness of specifying and receiving responses from the system. A format as close to English as possible both for input and output must be devised. The timing of responses must be closely allied to that of a human tutor. This amount of flexibility implies an on-line, real-time computer system.

D. A Graphic Terminal

A fast terminal with graphic capability is an intrinsic part of the system requirements. Students tend to get very impatient with the speed of present remote typewriter terminals. Furthermore, concentration on the word-by-word output as it is printed draws a student's attention to the minute detail (limited content) at the expense of the pattern or relationships (meaning). Since increasing

emphasis is being given to this particular area by computer manufacturers, at the present time we have no doubt that better terminals will be produced in the near future.

III. Operational Objectives

Our initial system possess some of the above mentioned qualities. Although our progress thus far is modest, the system is operational and can be utilized, studied, and improved following each experiment. Through the medium of the LISP language, a first-stage associative memory has been built which allows the student to ask questions of the type "what is," "where," "how" and "when," and also retrieve information on the relationship among concepts. (i.e., what is the relationship between A and B).

For subject matter we have chosen the fundamentals of accounting. The reason for our choice is that the subject is well-defined, and therefore easy to program. Its definitions are clear, and its interrelationships while complex enough to be challenging, may still be unambiguously stated. We must stress, however, that the basic system capabilities do not depend on the subject matter. In other words we are developing a potential general purpose system.

In order to allow the system to handle the dual roles of "lecturer" and "font of knowledge" two major interrelated capabilities are provided. First, the student has the opportunity to respond to the well-known programmed instruction mode of teaching. Secondly, and more uniquely, the system provides the student with the ability to ask questions of it and to receive appropriate responses.

We need not dwell long on the reasons for the programmed instruction phase, since a great deal has been written of the desirable attributes of programmed instruction. Among the advantages claimed most often are that:

1. Programmed instruction allows the slow student to proceed at his own pace without impeding the progress of those who can absorb the material faster.
2. As the student reads the material, he receives instantaneous feedback on his progress and therefore reinforcement of those concepts which appear to trouble him.
3. Programmed instruction provides a logical step-by-step approach to the subject matter and therefore enables the student to gain a firmer grasp of the material covered and also to retain it longer than other modes.
4. It provides opportunities for partitioning logically the material to be covered and providing decentralized control of the learning process.
5. Finally, programmed instruction relieves the instructor of the previously mentioned drudgery of teaching material that is very well known to him. As one of us has suggested elsewhere, "many times we find ourselves having to teach prerequisite courses, courses which we have taught for many years and from which we derive no challenge. If we succeed in programming the material which does not challenge us anymore, because we have succeeded in exhausting all the innovations we can achieve, then we will

> be free to introduce to the class material which is more challenging. Normally the latter material is full of ambiguity and of a speculative nature, and this challenges our ingenuity and imagination. This material, incidently, is something that the students also look for..."[1]

But programmed instruction alone can be both a boring and a frustrating method of learning. It is impossible to use many of the current programmed instruction systems without many times being put to sleep by the slow tedious pace of the material. One finds one's thoughts, at times, moving tangentially to the material but there is no way to escape. A fact, once forgotten, must be rescued by a search through the previous pages. (In other words, our own sense of haste, particular interests, and memory retention span may be quite different from that of the person who developed the particular programmed instruction text.)

A question-answer mode in our system tends to ameliorate or dispense with most of these problems. Three concepts underlie the basic question-answer ability.

> First, by using a simplified version of a semantic memory we have achieved a reasonable degree of flexible search. Accounting has some inherent formalism in its terminology so we can use "natural" accounting English language to ask

---

[1]Morton, M. S. S. and Zannetos, Z. S., "Efforts Toward an Associative Learning Instructional System," Proceedings of the IFIP Congress 68, Edinburgh, Scotland, August 1968. (Also MIT Working Paper No. 355-68.)

questions of the memory. Capitalizing on this inherent structure, we have been able to construct an adequate working system that appears to the user to be answering questions in natural English language.

Secondly, we have provided the ability for simple association. Concepts are stored in the semantic memory with links or pointers associating them to the other concepts to which they are principally related. These associations have been originally chosen on the a priori judgment of the designers of the system. As information on student associative methods is gathered (from data available on the stored trace), however, the associative paths are being reorganized to better meet the learning requirements of the users, and to incorporate new discoveries.

Thirdly, in similar fashion, we have provided for simple inference. The system traces through a series of levels in the semantic hierarchy in an attempt to link up the concepts raised by the user. This ability to relate aspects of accounting which have not been directly linked to each other by the programmer will be discussed later under the systems software section.

The student, however, is most often uninterested in the concepts behind the system. To him the question-answer ability means that:

1. If the current programmed instruction frame includes a word

or concept which he has previously learned, but has forgotten, he can merely ask a question of the system and have his memory refreshed on the spot. In this sense the system serves as an "exhaustive" indexing device and as a library.

2. If a programmed instruction frame suggests a tangential, but highly interesting, area of inquiry he may ask questions on this new area while the idea and his interest are still fresh. It is well understood that a student learns best when provided with material about a matter in which he is highly interested <u>at the particular time</u> at which his interest is blooming.

3. He may guide his own learning experience. We claim no greater ability to provide the single programmed instruction path which will exactly fit the thought progression of all students. Nor do we have as yet a system which restructures itself to appeal to the individual objectives of each particular user. What we now provide the student is flexibility. He can "escape" if he so wishes from the prescribed path and then return to the spot from where he left, at will. It is felt that this provides the student with a sense of control over his learning process. Rather than being merely led, he now has a chance to lead and to explore; he can become an <u>active</u> participant in the educational process rather than merely a passive recipient.

To us the above capabilities of the system mean that ultimately "better paths" may be developed for the user since the system allows teachers to trace the steps which the students take in their efforts to learn. The user "protocols" are actively analyzed for the recognition of significant patterns. As we learn more about how students actually learn, improvements may be made not only in the material used but also in the process of teaching itself.

With the above material as a background we will now turn to the hardware and software of our present system.

ASSOCIATIVE LEARNING HARDWARE

The associative learning programs are currently being run on the Compatible Time Sharing System at M.I.T. This is a very flexible time sharing system with a large number of programming languages available. The machine being used is an IBM 7094 computer with two banks of core each with 32,768 36-bit words of memory. During the operation of the system, the supervisor and the input-output routines reside in one bank and the time-shared programs are run in the other bank. This provides a system which, although somewhat outdated technically, renders reliable service and can handle up to 30 simultaneous users. Permanent storage is provided by an IBM 1302 disc unit with a capacity of 38 million 36-bit words. Disc space is allotted to users, like the associative learning project, to store programs and data.

The terminal that is being used for the project is a Computer Displays Inc. Advanced Remote Display Station (ARDS) the heart of which is a Tektronix 6 1/2 by 8 inch storage tube. The ARDS has the capability to store up to 50 lines of written material (80 characters to the line), or to accept graphical input in stored points and line segments. Although the terminal is capable of writing up to 500 characters per second, it is currently limited to 110 characters per second using a 202C type Dataphone and a local phone line. This writing speed, however, is more than adequate since the student is unable to read as fast as the machine is writing. The major disadvantage of this display is the fact that it is impossible to selectively erase the screen. Consequently, the material

must be presented to the student in pages and any graphical material that is required must be redrawn on each page. However, for written material such as described in this paper, it provides a cheap method of quickly presenting material to the student. There is, in addition on the terminal, a graphical input device of a "mouse" or "joystick." This allows the user to "point" to material when selecting alternatives and avoids the problem of always having to type responses.

ASSOCIATIVE LEARNING SOFTWARE

As previously discussed, the associative learning program operates in two major instructional modes: programmed instruction and question-answer. Because these modes are largely independent, they will be discussed separately, following a section on general file and start up procedures. Figure 1 provides as an overview, the systems files and the major divisions in the core storage "map."

Program Initialization

The associative learning program reads from three separate data files during its operation. These are:

"Text"	which contains all the questions for programmed instruction.
"Answer"	which contains all of the answer lists for the programmed instruction. The answers are formatted in blocks of ten answer lists per block.
"User Status"	which contains the names of students who have used the system and the next frame number for each student who has not completed the entire program.

When the student first starts the system, he is asked to type his name. With this name as an input, the "begin" function first looks in the "User Status" file under the student's name for the next question number. If it does not find the student's name, it assumes that he is a new student and assigns him the first question. Having retrieved the correct question the "begin" part of the program then opens the "Text"

and "Answer" files and reads in the proper block of answers. The student is then asked, "WOULD YOU LIKE TO ASK SOME QUESTIONS." If the student answers in the affirmative, the question-answer mode is entered. If he does not wish to ask a question, then the programmed instruction is started.

Programmed Instruction Mode

The programmed instruction mode (Figure 2) is supervised by a function called "test." "Test" starts by presenting the first (or next) question to the student. After presenting the question, "test" reads the student's answer and compares it with the answer list that has the same number as the question. If the answer given is correct, "test" prints the first correct answer on the list (the preferred answer) and the words "IS CORRECT" and waits for the student to ask for the next question. The number of this succeeding question is determined from a coded question number following each correct answer in the answer list.

If the answer to any question is incorrect, then "test" prints "YOU ARE INCORRECT--(preferred answer) IS THE CORRECT ANSWER." For incorrect answers, "test" also looks at the answer list for a specially encoded "question" which it presents to the question-answer program (described below). These special questions with the appropriate answers are then presented to the student as remedial material. After an incorrect answer, the next question is determined by a coded number at the end of the answer list.

"Test" then uses "Reopen" to determine if the designated next question precedes the one just completed by the student. If it does, the

files are closed and reopened. The following question is then obtained by reading "Text" and a simultaneous check is made to see if the answer to this question is in the current block of answers. If the answer is not present, then the current answer block is removed and "Answer" is read until the proper answer block is found.

If during the programmed-instruction mode, the student types the word "quest" between frames, then "test" stores the next frame number and calls the question answer program. If the student types the word "explain" between frames, then "test" gets and displays the block of remedial material for the last frame before going on to the next frame. If the word "stop" is typed between frames, then "test" writes the student's name and the next question number on "User Status" and prints the number of questions that the student answered correctly and incorrectly and ends the session.

Question-Answer Mode

The question-answer mode of the system is supervised by a function named "quest." The first operation that "quest" performs is to provide the student with the following message: "PLEASE ASK ANY QUESTION RELATED TO BASIC ACCOUNTING PRINCIPLES. WHEN YOU WISH TO HAVE THE PROGRAM ASK QUESTIONS AGAIN, TYPE THE WORD 'TEST' AND HIT THE RETURN KEY." Following this, "quest" reads the student's questions, encodes them and has the "analyzer" function answer them. (The methodology used for this is described below.) It then presents this answer to the student. If during this process the student enters the words "test" or "stop" on a single line, then "quest" returns to the "test" function for programmed instruc-

tion or closeout, respectively.

The Dictionary

To answer the student's questions, the "analyzer" uses an extensive dictionary of word forms that is stored in core during program execution. This dictionary uses the list structure and list processing capabilities of the LISP language to recognize four major types of word forms: keywords, supplementary words, question words and relation words.

The largest section of the dictionary is devoted to nouns and phrases which are used as keywords and which enable the system to understand the questions. For each keyword there is stored a "property-list" with up to six descriptions (properties) of the word stored on it. These properties are: (See Figure 3 for examples)

a. The definition - (DEF)

b. The use of the item - (USE)

c. A time context for the item - (WHEN)

d. A place context for the item - (WHERE)

e. Some examples of the item - (EXPL)

f. The relationship of the keyword to other keywords - (REL)

Like an ordinary dictionary the first five keyword properties (a through e) are followed by plain English descriptions of the noun or phrase in that context. The REL property, which is more complex, uses verb forms as relation words to relate the various keywords in the dictionary (see Figure 4 for a diagram of some of these relationships). Since the student may use synonyms or plurals, the system stores these supplementary words with

a pointer to the main keyword and uses the main keyword dictionary to answer questions which have been phrased by the use of supplementary words. The program must also recognize question words and phrases in order to understand the questions being asked. The question words which the program recognizes are "what," "what for," "how used," "when," "where," and "example" or "examples." Phrases are combinations of words which include keywords.

Question Answering Procedure

An overview of the question answering procedure is diagrammed in Figure 5. With the various word forms defined and available, the program proceeds as follows: "Quest" determines how many words of each type (keyword, supplementary word, question word, relational word) it can recognize in the student's question. It passes these words to the "analyzer" which in turn proceeds to classify the question as simple, complex, relational. Based on this classification, "analyzer" constructs one or more answers to the question and passes these back to "quest." If more than one answer has been constructed, "quest" presents the first one and stores the rest until the student asks for them in sequence by hitting the return key.

A simple question contains only one keyword and a question word. These questions are answered by simple dictionary lookup as follows: If the question word is "what" then the answer is the definition followed by the use. If the question word is "what for" or "how used" then the answer is the use property. If the question word is "when"

then the answer is the time context of the item. Similarly, "where" draws forth the place context and "example" or "examples" is answered by using the EXPL property. For example the question "What is an asset" causes "quest" to find the question word "what" and the keyword "asset." "Analyzer" then forms a single answer from the combination of the definition and use properties. The student is then given the answer "AN ASSET (OR FORM OF CAPITAL) IS TANGIBLE OR INTANGIBLE PROPERTY (A FORM OF RESOURCES) IN THE FIRM'S LEGAL POSSESSION.

A complex question contains two keywords with or without a question word. To answer this question the "analyzer" uses the relationships associated with the second word to find a relationship path to the first word. This path may contain several intermediate words as long as it leads from the second keyword to the first keyword. (See Figure 4. For example, if the keywords are "assets" and "wealth" the path would be "WEALTH IS ECONOMIC VALUE. VALUE MEASURES AN ASSET." This process is continued until all paths from the first to the second and from the second to the first are discovered and the relationships stored. In addition, if there is a question word in the question, a further answer is developed which is the answer to the simple question made up of the question word and the first keyword. So, for example if the question is "what is the title of an asset," "quest" finds the question word "what" and the two keywords "title" and "asset" and "analyzer" constructs two answers to be presented to the student. These are: (1) "AN ASSET IS IDENTIFIED BY A TITLE," (2) "A TITLE IS THE NAME OF AN ACCOUNT, WHICH

IDENTIFIES AN ACCOUNT." It can be seen from Figures 2 and 3 that the first answer is the relationship from the second keyword to the first, and the second answer is the DEF and USE properties of the first keyword "title" (i.e., the simple question answer). Of course, this is a rather trivial example of a complex question, but it can be seen how this principle can be used to provide some very complex relationships to the student.

The relational question has only one keyword, one or more relational words, and an optional question word. The "analyzer" answers this type of question by searching the keyword's REL property to determine if the relation words contained in the question are there. If a relationship is found then the answer provided is this relationship. For example, if the question is "how do you commit an asset," then "quest" recognizes no question words, the relational word "commit," and the keyword "asset." "Analyzer" then provides the relational answer "AN ASSET (OR FORM OF CAPITAL) IS COMMITTED BY AN EXPENDITURE." Since there is no question word in this question (i.e., no "used" with the "how") this is the only answer that is provided. If the question had been "what does an expenditure commit," then there would have been two answers: the relation between "expenditure" and "asset," and the definition of expenditure.

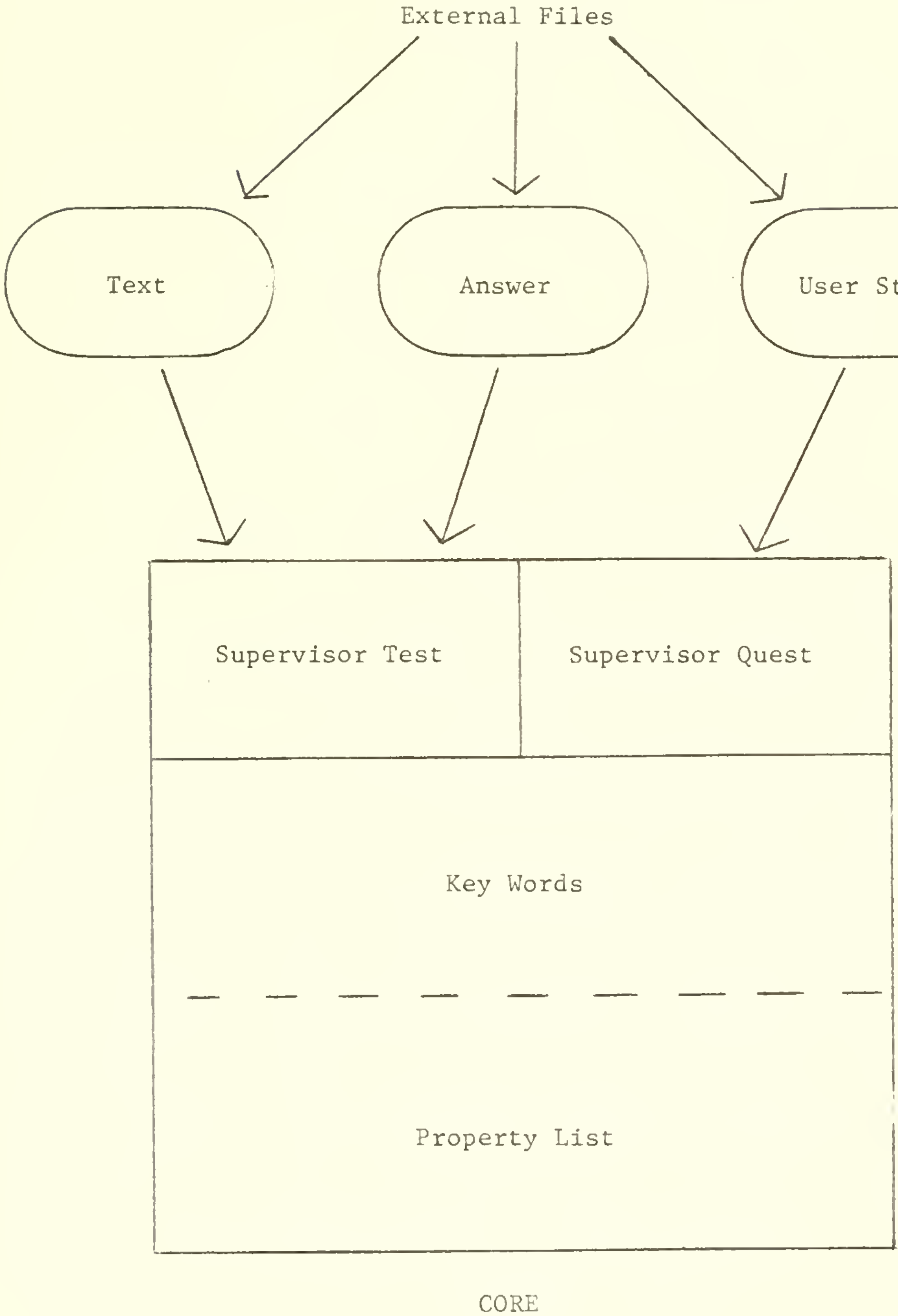
At least two words must be recognized in order to provide an answer. If the two words are not present or if the particular reference is not in the dictionary, then "quest" replies, "I DO NOT UNDERSTAND--

PLEASE REPHRASE YOUR QUESTION."

Throughout the operation of the ALP program, all interchange of information with the student is also written on the file named "Tutor Users." During programmed instruction operation the question number, the student's answer, and a grade of right or wrong are all encoded on one line. During the question-answer operation, each question and answer is printed verbatim as it is displayed to the student. This is the current methodology to allow the system developer or the teacher to grade the progress of the system or the student at a later date and also to view students' methods of structuring the learning process.

Figure 1

Storage Overview of the ALPS System



PROGRAMMED INSTRUCTION FLOW CHART

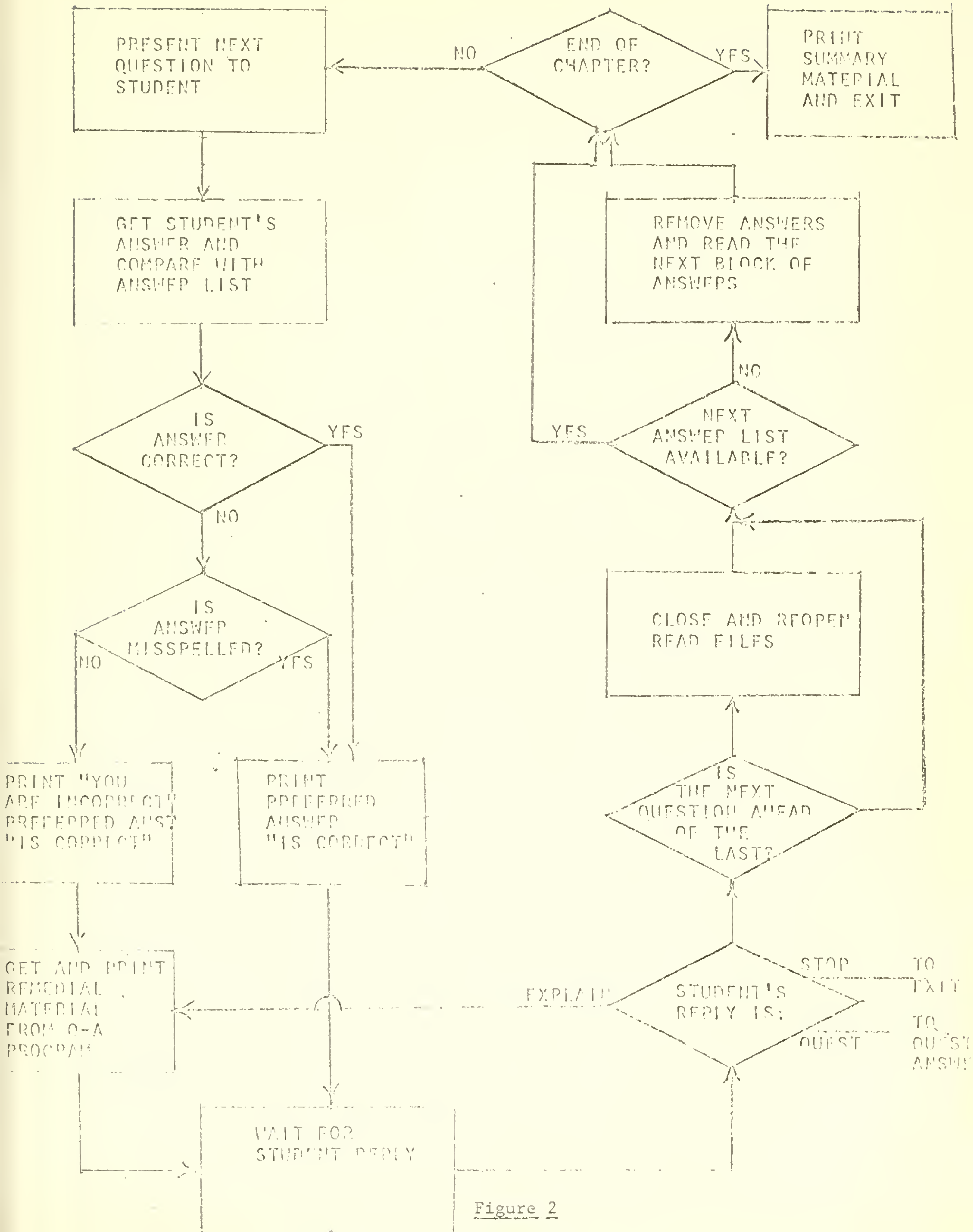


Figure 2

Figure 3

Examples of Dictionary Entries

ASSET - Symb = an asset (or form of capital)

DEF = is tangible or intangible property (a form of resources) in the firm's legal possession

USE = can be used to generate future value

WHEN = exists if there is value remaining at balance sheet time

WHERE = is noted on the left-hand side of a balance sheet

EXPL = can be cash, accounts receivable, inventory, buildings, or machinery

REL = (is a form of capital) (is decreased in value by credit) (is increased in value by debit) (is exemplified by inventory) (is economic value) (is identified by title) (is equivalent to property) (is committed by expenditure)

T-ACCOUNT - Symb = a t-account

DEF = is the simplest representation of a ledger account

USE = lists the impact of transactions on that entity. On the left-hand side are all the capital flows, the account received and on the right-hand side those that it gave.

WHEN = is used when all entries must be transferred from the journal to the ledger accounts prior to constructing a balance sheet

WHERE = is contained in the ledger

REL = (represents (written) account) (results in balance) (is contained in income statement) (is contained in balance sheet)

TITLE - Symb = a title

DEF = is the name of an account

USE = identifies an account

REL = (identifies asset)

VALUE - Symb = value

DEF = is the measurement, as amount in dollars, of the worth of an economic good. In accounting, value for assets is the same as cost.

REL = (measures asset) (is in terms of Unit of measurement)

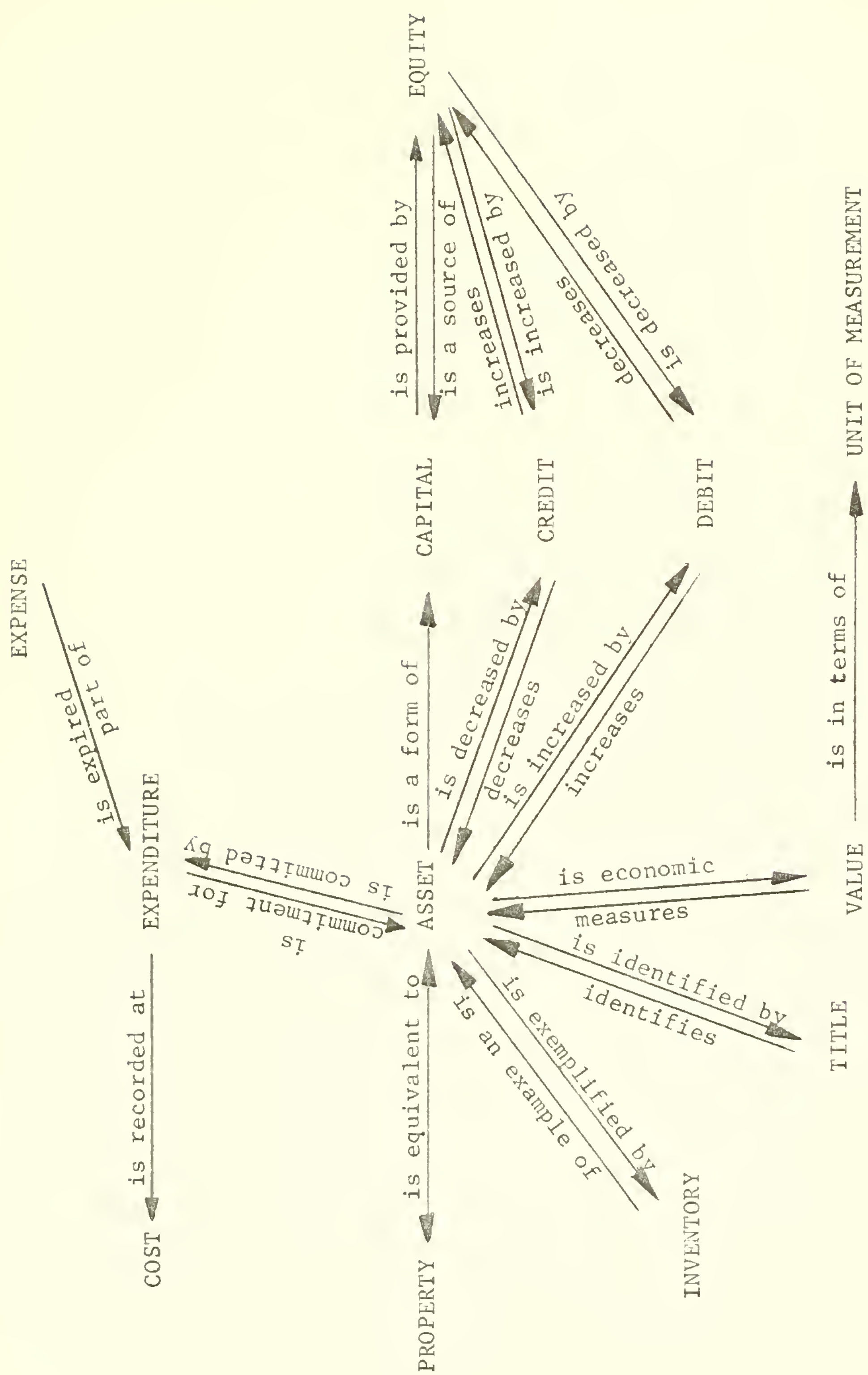
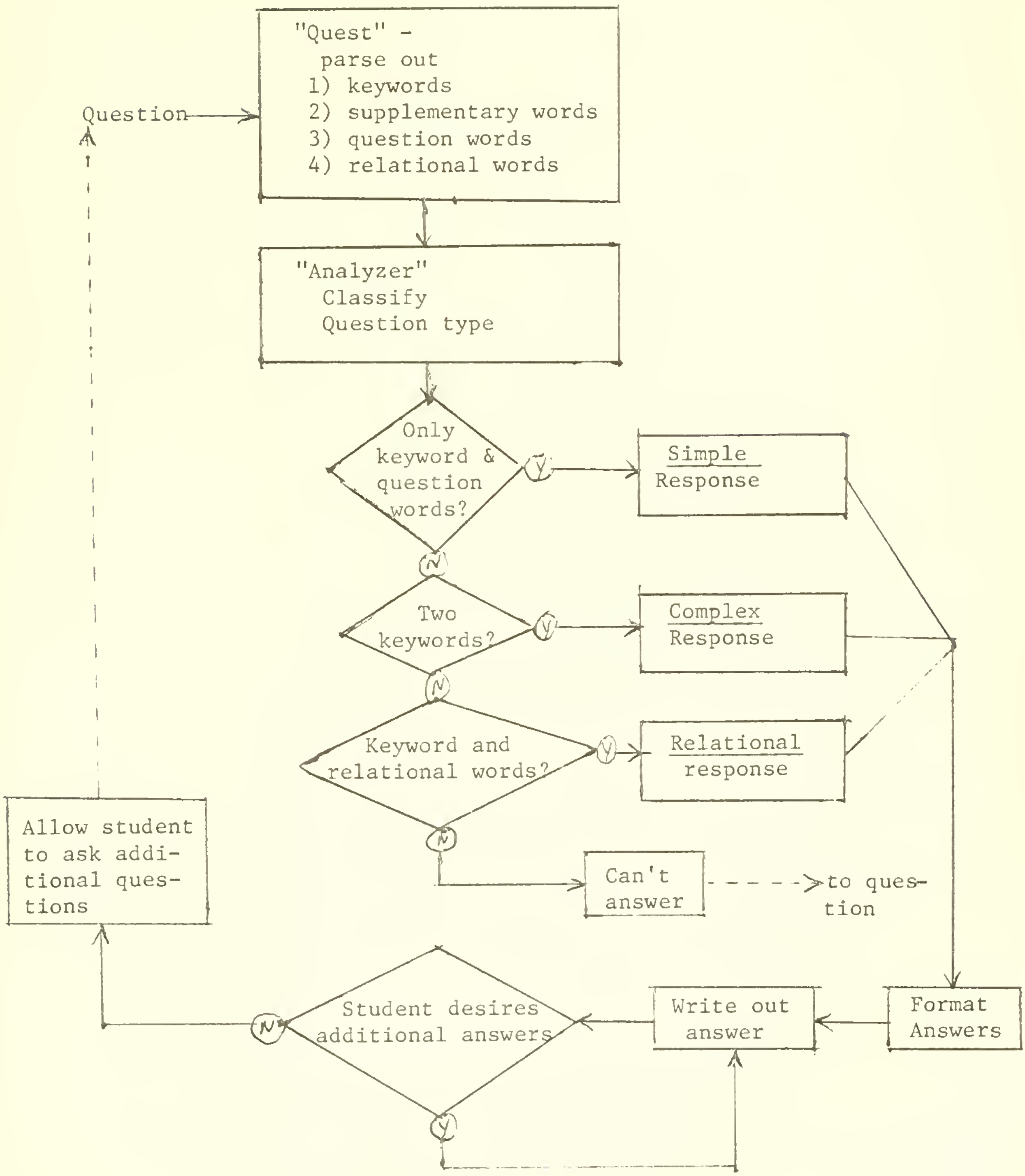

EXPENSE
is expired part of
EXPENDITURE
is recorded at
COST
is commitment for
is committed by
ASSET
is equivalent to
PROPERTY
is a form of
CAPITAL
is provided by
is a source of
EQUITY
is decreased by
decreases
CREDIT
increases
is increased by
is increased by
increases
DEBIT
decreases
is decreased by
is economic
measures
VALUE
is in terms of
UNIT OF MEASUREMENT
is identified by
identifies
TITLE
is exemplified by
is an example of
INVENTORY

Figure 4

Figure 5

Question-Answerer Data Flow



Date Due

DEC 05 '76	[redacted]	
FEB 05 '76	OCT 1 2 1983	
FEB 23 '78		
~~FEB 17 '77~~	FEB 14 1989	
SEP 24 '77		
NOV. 18		
[illegible]		
MAY 17 '78		
AUG 21 '80		
W		
[illegible] '82		
[redacted]		